Инженерный вестник Дона, №1 (2024)
ivdon.ru/ru/magazine/archive/n1y2024/8945

Проектирование механизмов сетевого обмена данными между интеллектуальными автономными роботами на базе распределённого key-value хранилища

А.В. Ермаков, Л.И. Сучкова

Алтайский государственный технический университет им. И.И. Ползунова, Барнаул

Аннотация: Рассматриваются вопросы функционирования коллектива агентов, объединённых через сеть. В работе рассмотрена архитектура распределённой системы обработки данных на основе оверлейной сети. Целью работы является разработка комплексного решения, обеспечивающего предобработку сигналов, надёжность хранения и передачи данных, высокую скорость принятия решений. Произведена интеграция распределённого key-value хранилища Etcd с реализованным клиентом оверлейной сети путём использования tun-интерфейса на каждом агенте и инкапсуляции пересылаемых данных IPv4-пакетов для дальнейшей отправки через андерлейную IPv6-сеть по протоколу, обеспечивающего надёжность доставки. Рассмотренное в работе решение позволяет автономно создавать быстро развёртываемые механизмы сетевого обмена, гарантировать доставку данных с приемлемой задержкой в условиях помех, джиттера, колебаний задержки доставки пакетов, недоставки пакетов.
Ключевые слова: оверлейная сеть, андерлейная сеть, IPv6, распределённое хранилище, распределённая база данных, tun-интерфейс, распределённая система обработки данных, интеллектуальный агент, групповая робототехника, автономный робот.

Введение

Существующие исследования в области искусственного интеллекта охватывают множество подобластей. Одной из таких подобластей является автономная робототехника, рассматривающая работу интеллектуальных агентов (далее — агент) в коллективе и передачу данных от одного агента другому.

Агентом является устройство, которое может воспринимать своё окружение через датчики и воздействовать на это окружение исполнительными механизмами. Поведение агента может быть описано функцией агента, которая отображает любую корректную последовательность собранных данных на некоторое действие.

В успешно действующих агентах задача вычисления функции агента разбивается на три подзадачи. Во-первых, при проектировании программного обеспечения разработчиками закладываются знания-правила функционирования окружения, с которым в будущем будет взаимодействовать агент. Используя эти знания, агент отсекает невозможные в текущем окружении ситуации. Во-вторых, происходят сами вычисления по выбору одного из доступных действий в условиях ограничений и предыдущего опыта взаимодействия с окружением. И, в-третьих, выполняется постобработка данных для окончательного принятия решения и модификации агентом своего поведения. Если агент в большей степени полагается на собственный опыт и собранные данные, чем на априорные знания своего проектировщика, то такой агент рассматривается как обладающий достаточной степенью автономности.

При выполнении обработки сигналов с датчиков, необходимо учитывать его зашумлённость и погрешность квантования. Реальные физические динамические системы носят непрерывный характер. Сигнал всегда является функцией от времени, значение аналогового сигнала определено для каждого момента времени и может принимать все значения из заданного диапазона измерений. На практике аналоговый сигнал преобразуется в дискретный, вне зависимости от вида сигнала результаты любых измерений физических величин искажены случайным шумом. Этот шум является следствием как природы измеряемых величин, так и погрешностей измерительных приборов. При измерении аналоговых величин добавляется шум квантования аналого-цифрового преобразователя (АЦП).

Возникает потребность в извлечении информации из зашумлённых измерений, так как присутствие нежелательного шума в наблюдаемых данных может повлиять на качество принимаемых агентом решений. Модели обработки зашумлённых данных рассмотрены в [1].

Успешность выполнения задач коллективом автономных роботов (агентов) зависит от надёжности коммуникаций между членами коллектива, а именно от гарантии доставки сообщения до исполнителя и получения ответа. Отсутствие информации о намерениях других агентов может привести к ситуации, когда отдельные агенты будут мешать друг другу выполнять поставленные задачи [2, 3]. Чтобы избежать этого, требуется глобальная связность, обеспечивающая надёжный обмен данными между агентами для глобального и локального планирования и выполнения локальных задач каждым агентом в последующем.

Скорость и качество принятия решений зависят, в том числе от актуальности и полноты собранных данных об окружении, в котором функционирует группа автономных роботов, следовательно, возникает потребность в надёжном распределённом хранилище данных, где автономные роботы могут сохранять собранные данные и поддерживать их в актуальном состоянии.

Таким образом, при проектировании механизма обмена данными требуется разработка комплексного решения, учитывающего предобработку сигналов, надёжность хранения и передачи данных, скорость принятия решений.

Интеллектуальные агенты

В данной работе мы предлагаем архитектуру распределённой системы обработки данных, которая может быть применима для организации надёжной коммуникации внутри коллектива автономных роботов (агентов).

Упрощённый алгоритм действий каждого агента состоит в следующем (рис. 1) [4, 5]:

- сбор данных от датчиков, которые возвращают информацию о текущем состоянии физического окружения;

- предварительная обработка данных с возможностью детектирования значимых сигналов среди полученных данных;

- сохранение обработанных данных в распределённом хранилище, получение данных из распределённого хранилища от других агентов;

- принятие решения на основе имеющихся актуальных данных;

- планирование последовательности действий;

- выполнение действий исполнительными устройствами, изменение состояния физического окружения.

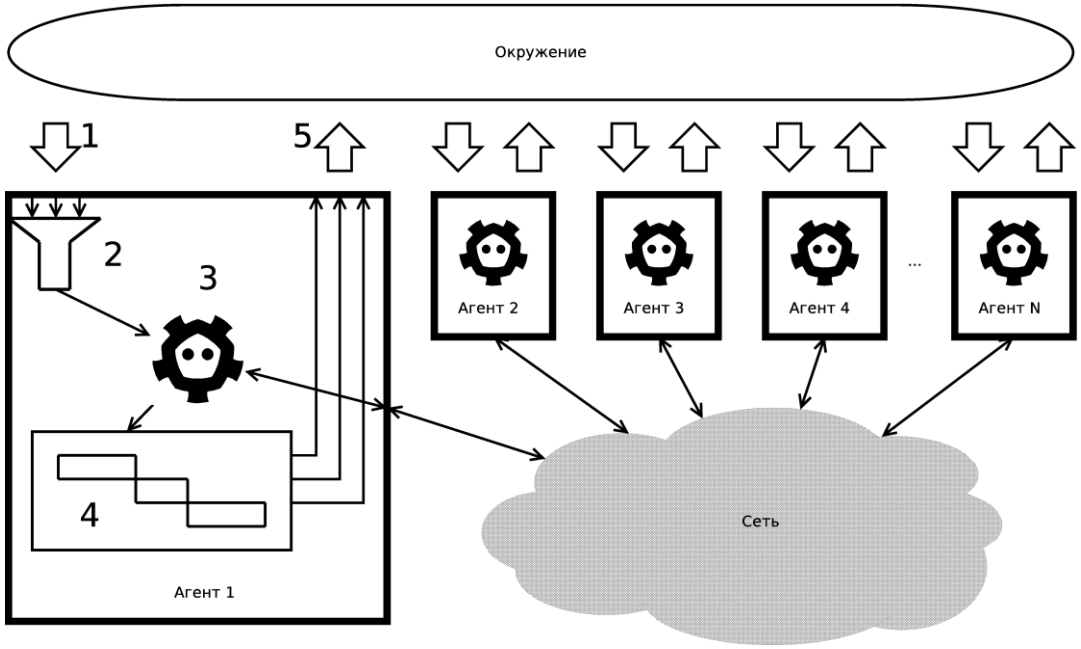

Рис. 1. — Схема функционирования коллектива агентов: (1) — получение данных от датчиков; (2) — предварительная обработка данных; (3) запись и чтение данных в распределённом хранилище; (4) — принятие решений и планирование; (5) — выполнение последовательности действий

Большинство вышеперечисленных пунктов алгоритма выполняется асинхронно и параллельно, например, получение данных от датчиков происходит с некоторой периодичностью. Поэтому вся хранимая информация в распределённом хранилище перед попаданием обогащается

метаинформацией, включающей отметку времени, когда было получено значение от датчиков, тип датчика, сырое (raw) значение данных, нормированное значение данных, уникальный идентификатор датчика, местоположение датчика на агенте, идентификатор агента. Хранимая метаинформация позволяет понимать, насколько актуальной является информация и источник получения информации. Алгоритмы предварительной обработки данных были рассмотрены в работе [6]. Остановимся более подробно на сетевом обмене агентов и использовании распределённого хранилища.

Сеть и связность сети играет главную роль в обеспечении коммуникации внутри коллектива агентов и доставки пакетов между узлами сети. В работе [7] нами был предложен прототип клиента оверлейной сети TwilightNet 0.1, который работал поверх андерлейной сети IPv4+UPD. Однако функционал перенаправления передаваемых данных в оверлейную сеть и обратно изучен не был, что значительно усложняет интеграцию оверлейной сети с распределённым хранилищем данных.

Распределённое хранилище

В качестве распределённого хранилища данных мы предлагаем использовать базу данных «ключ-значение» (далее — key-value хранилище) Etcd, которая хорошо себя показала в сравнении с аналогичными распределёнными хранилищами данных. Сравнение производительности записи значения в хранилища Etcd с другими аналогичными распределёнными key-value хранилищами и распределёнными системами управления базами данных (далее — СУБД) приведены в работе [8].

Кластер Etcd является распределённым in-memory хранилищем данных, то есть все данные хранятся в оперативной памяти, существует возможность создавать снимки данных (snapshots, снапшоты) и хранить их в постоянной

памяти, так и возможность восстанавливать снимки данных из постоянной памяти. Etcd для своего функционирования использует TCP-соединения и требует возможности установления соединений «все-со-всеми».

Оверлейная сеть

Помимо собственно передачи данных между узлами сети требуется обеспечение её связности, то есть, для любой пары узлов сети должен существовать маршрут, по которому может быть доставлен пакет от одного узла к другому. Высокая связность сети предполагает, что такая сеть надёжна.

Сеть всегда была самой инертной частью любой системы [9], производить изменения физической сети сложно, зачастую невозможно обновить аппаратные устройства в сети и произвести смену низкоуровневых протоколов обмена, таких как Ethernet или IEEE 802.11 (Wi-Fi) — второй (канальный) уровень модели OSI — или IP — третий (сетевой) уровень модели OSI.

В качестве одного из вариантов решения недостаточной гибкости базовых протоколов сети, наряду с виртуальными локальными сетями (virtual local area network, VLAN) и виртуальными частными сетями (virtual private network, VPN) существуют так называемые наложенные сети (overlay network, далее — оверлейная сеть).

Оверлейные сети абстрагируются от нижележащих протоколов, например, сеть может использовать различную среду передачи данных в разных сегментах гетерогенной сети. Единственным требованием к сетям, поверх которых работает оверлейная сеть, является наличие маршрута между подсетями. Такие сети позволяют расширить функционал сети, не затрагивая нижележащих базовых протоколов [10].

Достоинством оверлейной сети является то, что она работает поверх существующей физической сети, так называемой подлежащей сети (underlay network, далее — андерлейная сеть), и настраиваются только конечные узлы сети. Андерлейная сеть опирается на стандартные протоколы и технологии. Полезная нагрузка оверлейной сети сокрыта в передаваемых пакетах по сети, от андерлейной сети требуется лишь обеспечение связности сети. Транзитные узлы андерлейной сети не нуждаются в настройке и не учитывают наличие оверлейной сети и её топологии.

На рис. 2 приведён пример оверлейной сети с двумя конечными узлами.

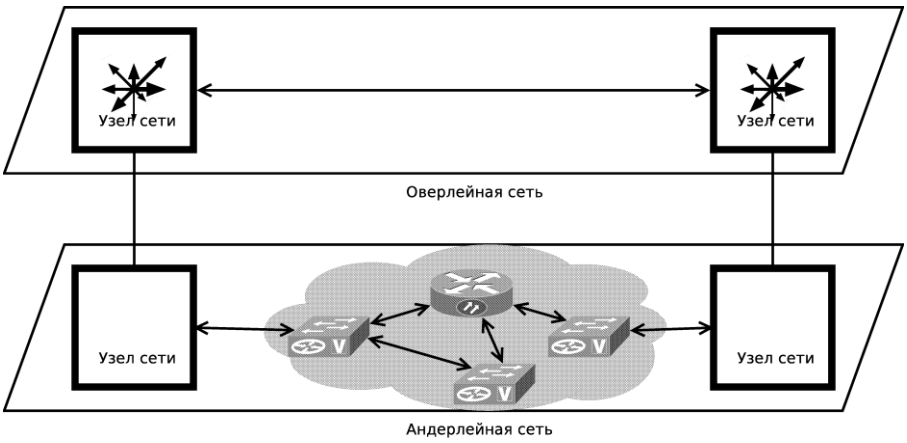

Рис. 2. — Оверлейная сеть, построенная поверх андерлейной сети

Первоначальным вариантом интеграции распределённого хранилища Etcd и оверлейной сети было использование правил перенаправления передаваемых данных средствами iptables (рис. 3а) [11, 12]. Но этот подход обладает рядом недостатков: сложность конфигурации правил iptables в полностью автоматическом режиме; необходимость доработки функциональности клиента оверлейной сети TwilightNet для возможности перенаправления передаваемых данных внутрь оверлейной сети и использование клиента в новой роли локального прокси.

Более правильным способом является поднятие на каждом конечном узле сети (агенте) собственного сетевого tun-интерфейса [13, 14], что позволит получить полную поддержку маршрутизации пакетов на уровне операционной системы агента и обеспечить доступ к содержимому пакетов на третьем (сетевом) уровне модели OSI (рис. 3б).

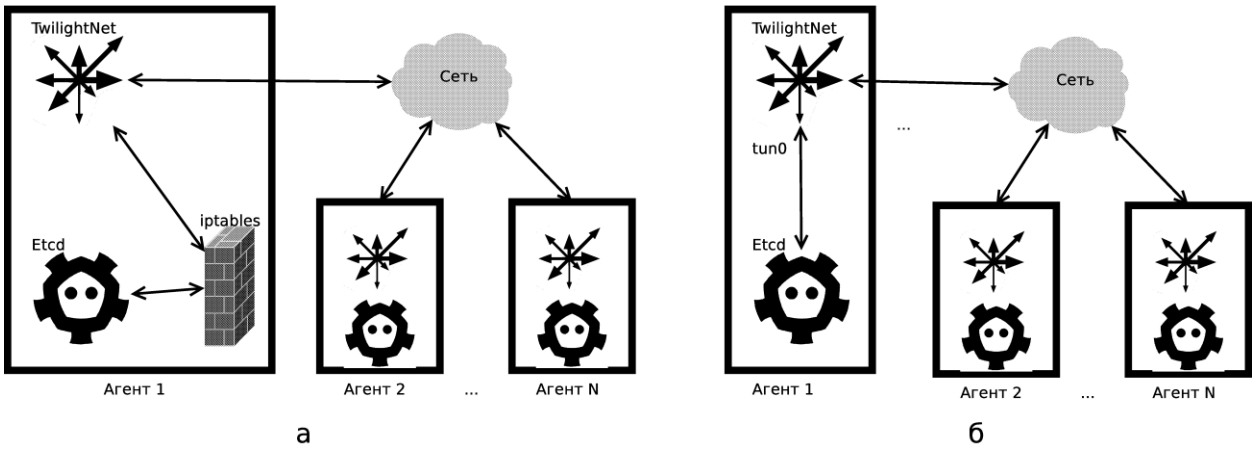

Рис. 3. — Доступ в оверлейную сеть на примере Etcd через: (а) правила iptables перенаправления передаваемых на локальный прокси; (б) tun-интерфейс клиента оверлейной сети TwilightNet 0.2

В версии клиента оверлейной сети TwilightNet 0.1, рассмотренной в [7], использовалась так называемая лавинная маршрутизация для рассылки изменений в структуре сети, в том числе информации о добавлении новых конечных узлов сети или о недоступности определённого узла сети. Хотя такой подход информирования об изменениях статуса сети позволял без особых затрат поддерживать в актуальном состоянии копию статуса сети на каждом из конечных узлов, но сама лавинная маршрутизация значительно нагружала каналы сети, каждый узел сети должен был самостоятельно отслеживать дубликаты пакетов, а также заниматься дальнейшей рассылкой широковещательных пакетов остальным соседям по сети. Дополнительно андерлейная сеть зачастую требовала нетривиальной настройки на стыках

между подсетями, так как широковещательные пакеты не могут распространяться дальше своего широковещательного домена, чаще всего это своя подсеть, где находится агент.

Таким образом, через андерлейную физическую сеть передаются все данные. С учётом специфики использования сети автономными агентами к андерлейной сети предъявляются следующие требования: требуется её быстрое развёртывание, быстрое достижение связности сети, настройка в автоматическом режиме. Для автономных роботов особенно важна надёжность функционирования. Нами предлагается использование сетевого протокола IPv6 в андерлейной сети, а выбор нижележащих протоколов и стеков технологий может быть любым. Например, проводной промышленный Ethernet либо распределённые беспроводные mesh-сети [15].

Андерлейная сеть

Андерлейная сеть была сконфигурирована с использованием только протокола IPv6. Использование разных стеков протоколов для оверлейной и андерлейной сетей (IPv4 и IPv6, соответственно) позволило гарантированно разграничить пересылаемые данные между этими двумя сетями (см. рис. 4).

Одним из преимуществ использования протокола IPv6 является автонастройка local-scope адресов, причём для одного и того же узла сети будет сохраняться постоянство адреса. Принципы автонастройки описаны в стандарте RFC 4862 [16]. Кроме самостоятельного назначения IPv6-адреса узлам сети, что уже само по себе позволяет начать передавать пакеты с данными внутри подсети fe80::/10, IPv6 предоставляет протокол обнаружения соседей (Neighbor Discovery Protocol, NDP). После назначения IPv6-адреса каждый узел проверяет свой адрес на уникальность и по NDP, который является частью протокола ICMPv6, многоадресной (multicast) рассылкой распространяет сведения о подключении нового узла к сети.

Помимо этого, NDP может работать по принципу «запрос-ответ», на пакет-запрос в многоадресной рассылке отвечает каждый узел, который получил пакет.

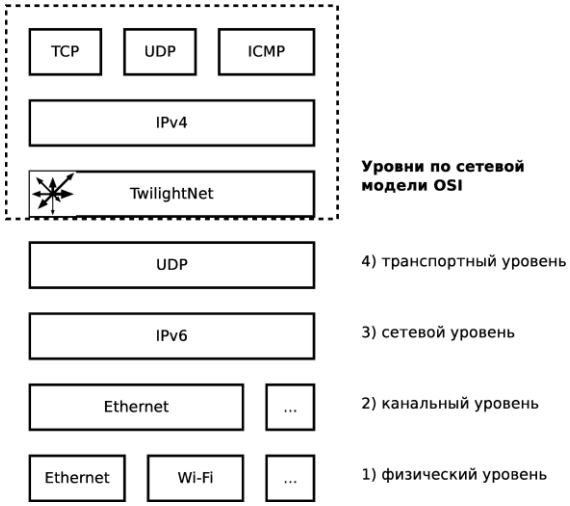

Рис. 4. — Инкапсуляция передаваемых данных по оверлейной сети в IPv6-пакет андерлейной сети

Операционная система отслеживает список соседей, поэтому отпадает потребность в дополнительном функционале сбора и поддержания в актуальном состоянии статуса андерлейной сети.

Клиенту оверлейной сети остаётся только подсоединиться к клиентам на других узлах сети, поднять собственно оверлейную сеть и открыть локально tun-интерфейс, через который будет осуществлено подключение к распределённому хранилищу Etcd.

Итоговая архитектура механизмов сетевого обмена при использовании распределённого хранилища

В результате зоны ответственности в архитектуре распределённой системы обработки данных агентов выглядят следующим образом:

- андерлейная сеть IPv6+UDP обеспечивает назначение адресов узлам сети, обнаружение соседей, связность сети и доставку пакетов между узлами сети;

- оверлейная сеть TwilightNet 0.2 обеспечивает подключение новых узлов оверлейной сети путём пересылки служебных сообщений, создаёт уникальный внутренний IPv4-адрес оверлейной сети, инкапсулирует передаваемые данные в собственный протокол на основе UDP с увеличенной надёжностью доставки и минимальными круговыми задержками сети;

- распределённое key-value хранилище Etcd обеспечивает хранение актуальных данных и их доставку между узлами оверлейной сети, горизонтальную масштабируемость, высокую производительность благодаря хранению данных в оперативной памяти и сохранению только последнего состояния каждого параметра. При необходимости получения ретроспективных данных агенты могут самостоятельно организовать получение срезов данных из Etcd и хранение их, например, в локальной СУБД;

- движок предварительной обработки данных осуществляет обработку данных в реальном времени, помещение обработанных данных в распределённое хранилище данных;

- датчики агента производят получение данных об окружении агента с некоторой периодичностью, далее данные обогащаются метаинформацией, такой как отметкой времени, и передаются движку предварительной обработки данных;

- подсистема принятия решений по основе правил забирает актуальные данные из распределённого хранилища данных и, руководствуясь полученными данными от локальных датчиков агента, а также данными от других агентов, принимает решения о выполнении действий. Осуществляет планирование как локальное, затрагивающее только одного агента, так и

глобальное — в последнем случае среди коллектива агентов путём достижения консенсуса выбирается лидер, который и осуществляет глобальное планирование и координацию действий коллектива агентов или части коллектива агентов;

- исполнительные устройства выполняют запланированные действия согласно заданным таймингам (времени выполнения).

Большинство подсистем агента работает асинхронно, отсутствуют глобальные блокировки, что на порядки увеличивает производительность распределённой системы в целом. Обмен данными между подсистемами агента и между узлами сети (другими агентами) осуществляется только через распределённое хранилище.

Запуск Etcd и клиента TwilightNet 0.2 осуществляется через подсистему управления службами systemd на каждом конечном узле сети. Для написания программного кода использовался язык программирования Rust и библиотека tokio-tun [17], позволяющая создавать сетевые tun-интерфейсы на операционных системах с ядром Linux. Тестирование проведено на операционной системе Debian 12.

Выводы

В данной работе нами была предложена архитектура распределённой системы обработки данных, которая может быть применима для организации коммуникации между агентами посредством оверлейной сети. За прототип был взят существующий клиент оверлейной сети TwilightNet 0.1, результатом работы стало расширение функционала клиента оверлейной сети, в программный клиент добавлено создание tun-интерфейса, добавлены разбор входящих IPv4-пакетов для передаваемых данных и инкапсуляция пересылаемых данных для дальнейшей отправки через андерлейную сеть. Дополнительно в клиент добавлена реализация примитивного программного

межсетевого экрана (firewall), чтобы пропускать внутрь оверлейной сети только TCP-пакеты от распределённого key-value хранилища Etcd, тем самым уменьшив нагрузку на нижележащую андерлейную сеть.

Рассмотренное решение позволяет автономно создавать быстро развёртываемые механизмы сетевого обмена, гарантировать доставку данных с приемлемой задержкой в условиях помех, джиттера, колебаний задержки доставки пакетов, недоставки пакетов.

Литература

1. Einicke G. Smoothing, filtering and prediction: Estimating the past, present and future (2nd ed), Amazon Prime Publishing, 2019, 380 p, ISBN 978-0-6485115-0-2.

2. Гайдук А.Р., Капустян С.Г., Шаповалов И.О. Алгоритм управления движением группы мобильных роботов в условиях неопределенности // Инженерный вестник Дона, Ростов-на-Дону, 2018, № 3. URL: ivdon.ru/ru/magazine/archive/n3y2018/5221

3. Филяев Г.А., Вилисов В.Я. Алгоритм концептуального проектирование системы мониторинга объекта коллаборативной мультиагентной робототехнической системой // Инженерный вестник Дона, Ростов-на-Дону, 2021, № 5. URL: ivdon.ru/ru/magazine/archive/n5y2021/6965

4. Ermakov A., Suchkova L. Development of Data Exchange Technology for Autonomous Robots Using a Self-Organizing Overlay Network, IEEE, 2019 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon), Vladivostok, 2019, pp. 1-5, DOI: 10.1109/FarEastCon.2019.8934727, URL: ieeexplore.ieee.org/abstract/document/8934727/

5. Ermakov A., Suchkova L. Multi-agent Planning of Autonomous Robots Based on Declarative Rules, IEEE, 2020 International Multi-Conference on

Industrial Engineering and Modern Technologies (FarEastCon), Vladivostok, 2020, pp. 1-5, DOI: 10.1109/FarEastCon50210.2020.9271331, URL: ieeexplore.ieee.org/abstract/document/9271331/

6. Ermakov A., Suchkova L. Pre-processing of observation data of intelligent agents using real-time causal filters, AIP Publishing, Almaty, Kazakhstan, 2023, 1 (2948), DOI: 10.1063/5.0165237, URL: pubs.aip.org/aip/acp/article/2948/1/020044/2920545

7. Ермаков А.В., Сучкова Л.И. Разработка алгоритмов для надежного обмена данными между автономными роботами на основе принципов самоорганизующейся сети // Надежность, Москва, 2020, № 2(20). С. 35-42, DOI: 10.21683/1729-2646-2020-20-2-35-42, URL: dependability.ru/jour/article/view/372

8. Ермаков А.В., Сучкова Л.И. Исследование структурных особенностей распределённых хранилищ для группы автономных роботов // Измерение, контроль, информатизация, Барнаул, 2023. С. 159-165

9. АДСМ1. Виртуализация сети, 2019. URL: linkmeup.ru/blog/1254/

10. Clark D. et al. Overlay Networks and the Future of the Internet, Communications and Strategies. 2006, 63, pp. 109-130.

11. Purdy G.N. Linux iptables Pocket Reference: Firewalls, NAT & Accounting. O'Reilly Media, Inc., 2004, p. 91, ISBN: 0-596-00569-5

12. Bauer M. Paranoid penguin: Using iptables for local security // Linux Journal, 2002, pp. 14-20, URL: dl.acm.org/doi/fullHtml/10.5555/563953.563967

13. Sanyal P. et al. An experience of implementing IPv6 based data retrieval system for Wireless Sensor Networks // 2012 International Conference on Recent Advances in Computing and Software Systems, IEEE, 2012, pp. 154-157, DOI: 10.1109/RACSS.2012.6212715, URL: ieeexplore.ieee.org/abstract/document/6212715/

14. Shuba A. et al. Antmonitor: Network traffic monitoring and real-time prevention of privacy leaks in mobile devices // Proceedings of the 2015 Workshop on Wireless of the Students, by the Students, & for the Students, ACM, 2015, pp. 25-27, DOI: 10.1145/2801694.2801707, URL: dl.acm.org/doi/abs/10.1145/2801694.2801707

15. Bicket J. et al. Architecture and evaluation of an unplanned 802.11 b mesh network, ACM, Proceedings of the 11th annual international conference on Mobile computing and networking, 2005, pp. 31-42. DOI: 10.1145/1080829.1080833, URL: dl.acm.org/doi/abs/10.1145/1080829.1080833

16. RFC 4862 IPv6 Stateless Address Autoconfiguration, IETF Datatracker, 2007, URL: datatracker.ietf.org/doc/html/rfc4862

17. Asynchronous allocation of TUN/TAP devices in Rust using tokio, 2023, URL: github.com/yaa110/tokio-tun

18. Benson K.E. et al. Resilient overlays for IoT-based community infrastructure communications, IEEE, 2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI), 2016, pp. 152-163, DOI: 10.1109/IoTDI.2015.40, URL: ieeexplore.ieee.org/abstract/document/7471359/

19. Лавров Д.Н. Принципы построения протокола гарантированной доставки сообщений // Математические структуры и моделирование, Омск, № 4 (48), С. 139-146, DOI: 10.25513/2222-8772.2018.4.139-146

References

1. Einicke G. Smoothing, filtering and prediction: Estimating the past, present and future (2nd ed), Amazon Prime Publishing, 2019, 380 p, ISBN 978-0-6485115-0-2.

2. Gajduk A.R., Kapustjan S.G., Shapovalov I.O. Inzhenernyj vestnik Dona, 2018, № 3. URL: ivdon.ru/ru/magazine/archive/n3y2018/5221

3. Filjaev G.A., Vilisov V.Ja. Inzhenernyj vestnik Dona, 2021, №5. URL: ivdon.ru/ru/magazine/archive/n5y2021/6965

4. Ermakov A., Suchkova L. IEEE. Vladivostok, 2019, pp. 1-5. DOI: 10.1109/FarEastCon.2019.8934727, URL: ieeexplore.ieee.org/abstract/document/8934727/

5. Ermakov A., Suchkova L. IEEE. Vladivostok, 2020, pp. 1-5. DOI: 10.1109/FarEastCon50210.2020.9271331, URL: ieeexplore.ieee.org/abstract/document/9271331/

6. Ermakov A., Suchkova L. AIP Publishing. Almaty, Kazakhstan, 2023, 1 (2948). DOI: 10.1063/5.0165237, URL: pubs.aip.org/aip/acp/article/2948/1/020044/2920545

7. Ermakov A.V., Suchkova L.I. Nadezhnost'. Moskva, 2020, 2(20), pp. 35-42. DOI: 10.21683/1729-2646-2020-20-2-35-42, URL: dependability.ru/jour/article/view/372

8. Ermakov A.V., Suchkova L.I. Izmerenie, kontrol', informatizacija. Barnaul, 2023. pp. 159-165

9. ADSM1. Virtualizacija seti [AfLO1. Network virtualization], 2019, URL: linkmeup.ru/blog/1254/

10. Clark D. et al. Communications and Strategies. 2006, 63, pp. 109-130.

11. Purdy G.N. Linux iptables Pocket Reference: Firewalls, NAT & Accounting. O'Reilly Media, Inc., 2004, p. 91, ISBN: 0-596-00569-5

12. Bauer M. Paranoid penguin: Using iptables for local security [Linux Journal]. 2002, pp. 14-20. URL: dl.acm.org/doi/fullHtml/10.5555/563953.563967

13. Sanyal P. et al. IEEE. 2012, pp. 154-157. DOI: 10.1109/RACSS.2012.6212715, URL: ieeexplore.ieee.org/abstract/document/6212715/

14. Shuba A. et al. ACM. 2015, pp. 25-27. DOI: 10.1145/2801694.2801707, URL: dl.acm.org/doi/abs/10.1145/2801694.2801707

15. Bicket J. et al. ACM. 2005, pp. 31-42. DOI: 10.1145/1080829.1080833, URL: dl.acm.org/doi/abs/10.1145/1080829.1080833

16. RFC 4862 IPv6 Stateless Address Autoconfiguration. 2007, URL: datatracker.ietf.org/doc/html/rfc4862

17. Asynchronous allocation of TUN/TAP devices in Rust using tokio, 2023. URL: github.com/yaa110/tokio-tun

18. Benson K.E. et al. IEEE. 2016, pp. 152-163. DOI: 10.1109/IoTDI.2015.40, URL: ieeexplore.ieee.org/abstract/document/7471359/

19. Lavrov D.N. Matematicheskie struktury i modelirovanie. Omsk, 4 (48), pp. 139-146. DOI: 10.25513/2222-8772.2018.4.139-146

Дата поступления: 27.11.2023

Дата публикации: 6.01.2024